SOURCE CODE APPENDIX

```
;=====================================================================
;                         Software UART for SPI-to-RS232
;                      for National Semiconductor's COP8SAx
;
;                           Rev 0.1, February 20, 1998
;
;         > Configured for COP8SAC @ 10MHz
;         > Hardware target = COP8-EVAL-HI01 (COP8 Evaluation Board)
;         > Uses "HyperTerminal" under Windows 95
;
;                            by:  Steven Goldman
;                                 National Semiconductor
;                                 Senior Field Applications Engineer

;
;
;
         .TITLE SPI-232
         .CHIP  8SAC
         .SECT  MAIN,ROM,ABS=0
;
;
;
;
;
;DECLARATIONS:

            PORTFD   = 0x94          ; PORTF Data Reg
            PORTFC   = 0x95          ; PORTF Config Reg
            PORTFP   = 0x96          ; PORTF Register (Input Only)
            DIPS     = 0x96          ; Dip Switches
            LEDS     = 0xDC          ; LED's
                                     ;
            TAURLOB  = 0E6           ; Timer B Reload, Low
            TAURHIB  = 0E7           ; Timer B Reload, High
            TIMERLO  = 0EA
            TIMERHI  = 0EB           ;
            TAURLO   = 0EC           ; Timer A Reload, Low
            TAURHI   = 0ED           ; Timer A Reload, High
                                     ;
            CNTRL    = 0EE
            PSW      = 0EF
            PORTLD   = 0D0
            PORTLC   = 0D1
            PORTLP   = 0D2
            PORTGD   = 0D4
            PORTGC   = 0D5
            PORTGP   = 0D6
            R0       = 0F0
            R1       = 0F1
            TRUN     = 4
            TPND     = 5
            RECREG   = 020          ;REG TO HOLD RECEIVED DATA.
            STKPTR   = 0xFD           ; Stack Pointer
```

```
;=====================================================================
; RECEIVE PORTION
;
;    1/9600 BAUD = 104 uSEC/BIT DECIMAL = 0068 HEX
;    1/2 BIT TIME IS = 52 uSEC = 52 DECIMAL = 0034 HEX.
;
;
;

START:     LD PORTFC, #0x00        ; Setup PortF as INPUT
           LD A, DIPS
           IFEQ A, #0x00           ; Dislay Revision Number
           JMP REVNUM              ;
                                   ;
           IFEQ A, #0x01           ; Receive Routine
           JMP RECROUT             ;
                                   ;
           IFEQ A, #0x02           ; Transmit Routine
           JMP CALLXMIT            ;
                                   ;
           IFEQ A, #0x03           ; Toggles RXD line
           JMP DEBUG1              ;
                                   ;
           IFEQ A, #0x04           ; Transmit "N"
           JMP SEND_N              ;
                                   ;
           LD A, #0xFF             ; Error Trap
           JSR ATOLEDS
           JMP HERE
                                   ;
                                   ;
                                   ;
;
;-------------------------------------------------------------------
;
REVNUM:    LD A, #0x17             ;
           JSR ATOLEDS             ;
           JMP HERE                ;
;
;-------------------------------------------------------------------
;
DEBUG1:    JSR ATOLEDS             ; Displays the Routine Number (3)
           RBIT 0, PORTLC          ; Make sure it is input pin
           SBIT 1, PORTLC          ; Configure RXD pin as OUTPUT
           LD B, #PORTLD
TOGGLE:    SBIT 1, [B]
           RBIT 1, [B]
           JP TOGGLE
;
;---------------------------------------------------------------------
;
RECROUT:   JSR ATOLEDS
           RBIT 0, PSW             ; Disable all interrupts.
           LD SP, #02F
              RC
           LD PORTGC ,#0x08        ; SET UP G1,& G2 AS INPUTS.
           LD PORTLC, #0x0E        ; Set up L0 as input, L1/L3 as output.
             SBIT 1, PORTLD
             RBIT 3, PORTLD
                                   ;
```

```
                                     ;
STRTRX:     CLRA
            RBIT 3, PORTLD
             RBIT TRUN, CNTRL         ; Make sure timer1 is off.
            LD TIMERLO, #0x0E        ; Load Half timer LB
             LD TIMERHI, #0x00        ; Load Half timer HB
SETIMR:     LD TAURLO, #0x62         ; Load Baudrate LB
             LD TAURHI, #0x00         ; Load Baudrate HB
            LD TAURLOB, #0x00        ;
            LD TAURHIB, #0x00        ;
             LD CNTRL, #0xA0
             LD R1, #0x08             ; (n-1) Data bits=8
                                     ;
IDLE:       IFBIT 0, PORTLP
             JP TRIGGER
             JP IDLE
                                     ;
                                     ;
             RBIT 2, PORTLD
TRIGGER:    SBIT 3, PORTLD
                                     ;
CHECK:      SBIT TRUN , CNTRL        ; Start Timer
            RBIT TPND , PSW          ; Reset Interrupt pending flag
CHECK0:     IFBIT TPND , PSW         ; Test Int flag
            JP CONTST
            JP CHECK0
CONTST:     RBIT TRUN, CNTRL         ; Stop the timer
             SBIT TRUN, CNTRL         ; Start the timer
            RBIT TPND , PSW          ; Reset Interrupt Pending flag
             IFBIT 0, PORTLP          ; Test for valid Start Bit
             JP VALSTART
             JP STRTRX
                                     ;
                                     ;
                                     ;
VALSTART:   SBIT 2, PORTLD
             RBIT 2, PORTLD
                                     ;
                                     ;
                                     ;
RECEV:                               ;
                                     ;
CHECK1:     IFBIT TPND, PSW          ; Receive bit in the middle
             JP CONT
             JP CHECK1
CONT:       RBIT TRUN, CNTRL         ; Stop the timer
             SBIT TRUN, CNTRL         ; Start the timer
             RBIT TPND, PSW           ;
            SBIT 2, PORTLD           ; Sampling pulse, per bit
            RBIT 2, PORTLD           ;
                                     ;
            LD A, RECREG             ; Load receive buffer
            SC                       ; Assume this was at Ground, then "1"
            IFBIT 0, PORTLP          ; If at +5VDC, then "0"
            RC                       ; Reset Carry is skipped if "1"
            RRCA                     ; Either way, rotate Right
            X A, RECREG              ; Store as latest value
            DRSZ R1                  ; Are we done yet?
            JP RECEV                 ; No...get more
```

```
          ;
FINISH:   SBIT 3, PORTLD          ; Golly! We are almost done
          LD A, RECREG            ; Display byte
          JSR ATOLEDS
          RBIT 3, PORTLD          ; Trigger scope (end of frame)
          JP STRTRX               ; Go get more
;
;
;
;======================================================================
ATOLEDS:            ; Value must be in Accumulator
                          ;   Since 1=LED Off, "A" must
                          ;   become NOT A (or /A).  Inverted
                          ;   value is then displayed.  Flow
                          ;   returns to caller.
                          ;
                          ;
        IFEQ A, #0X0D            ; If carriage return (0x0D), return.
        RET
        XOR A, #0xFF             ; Invert each bit
        LD B, #LEDS              ;
        X A, [B]                 ; Transfer /A to LED's
        LD A, LEDS
        XOR A, #0xFF
        RET                      ;
;
;======================================================================
;
HERE:     JMP HERE                ; Subroutine used to wait
                                        ;   for Reset
                                ;
                                ;
                                ;
;======================================================================
;
;
;     TRANSMISSION PORTION
;     ----------------------
;     Generic Calling Routine
;
XMIT:                             ; Soft UART Transmit routine
                                ;   Uses L.1 as an output
                                ;   Assumes L.0 is input
                                ;   Supports Half-duplex mode
                                ;
          SBIT 3, PORTLC          ; Set TRIGGER (L.3) as output
          SBIT 1, PORTLC          ; RXD (send to PC)
          RBIT 0, PORTLC          ; TXD (from PC)
          LD TIMERLO, #0x62       ; Setup Timers
          LD TIMERHI, #0x00       ;
          LD TAURLO, #0x62        ;
          LD TAURHI, #0x00        ;
          LD TAURLOB, #0x00       ;
          LD TAURHIB, #0x00       ;
          LD CNTRL, #0xA0         ;
                                ;
          LD R1, #0x08            ; Set for 8 data bits
                                ;
          RBIT 3, PORTLD          ; Set TRIGGER (L.3) LOW for frame sync
          SBIT 3, PORTLD         ; Set TRIGGER (L.3) HIGH for frame sync
          RBIT 1, PORTLD         ; Transmit Start Bit (0)
          JSR WFOBT              ; Wait For One Bit Time
                                 ;
MOREBITS: RRCA                   ; More next bit to "CARRY"
                                 ;
          RBIT 1, PORTLD         ; Assume we XMIT "0"
          IFC                    ; Are we wrong?
          SBIT 1, PORTLD         ; Sorry, XMIT "1"
          JSR WFOBT              ; Either way, wait
          DRSZ R1
          JMP MOREBITS
                                 ;
SENDSTOP: SBIT 1, PORTLD         ;
          JSR WFOBT              ;
          RET                    ; Return to calling routine
;
;------------------------------------------------------------------
;
WFOBT:    SBIT TRUN, CNTRL       ; Wait For One Bit Time
          IFBIT TPND, PSW
          JP BT_DONE             ; Get ready for next one
          JP WFOBT
BT_DONE:  RBIT TPND, PSW         ; Reset Timer
          RET                    ; Return to Calling Routine
;
;------------------------------------------------------------------
;
CALLXMIT: LD LEDS, #0xF8         ;
          LD A, #'C'             ; Transmit "COP8-"
          JSR XMIT               ;
          LD A, #'O'             ;
          JSR XMIT               ;
          LD A, #'P'             ;
          JSR XMIT               ;
          LD A, #'8'             ;
          JSR XMIT               ;
          LD A, #'-'             ;
          JSR XMIT               ;
          JMP CALLXMIT           ; Do it again, & again, & again...
                                 ;
                                 ;

;
;------------------------------------------------------------------
;
SEND_N:   LD LEDS, #0xFB         ;
AA:       LD A, #'N'             ;
          JSR XMIT               ;
          JMP AA                 ;
;
;------------------------------------------------------------------
;
;
         .END START
```